REMARKS

This responds to the Office Action mailed May 1, 2009 for the above application in which claims 144, 145, 155-158, 160, 164-165, 167-172 and 1615 are pending. Reconsideration of the application and claims in light of the following is requested.

Response to Section 112 Rejection

Claims 144-145, 155-158, 160, 164-165 and 167-172 were rejected under 35 U.S.C. §112, ¶2 as indefinite. Claim 144 has been amended to better clarify the elemental processing resources and their configuration and, in doing so, the rejection is obviated. Accordingly, it is respectfully submitted that claims 144-145, 155-158, 160, 164-165 and 167-172 are definite and withdrawal of the indefiniteness is respectfully requested.

Response to Section 103 Rejections

Claims 144, 155-156, 164-165 and 167-170 have been rejected for obviousness over a combination of the previously cited Bonola and Bridges references. Claim 144-145 and 171-172 have been rejected for obviousness over a combination of the previously cited Butterworth and Bridges references. Claim 157-158, 160 and 1615 have been rejected for obviousness over a combination of the previously cited Bonola, Bridges and Mohamed references.

Applicant respectfully traverses the obviousness rejections for the reasons set forth below. However, as a preliminary matter, it is respectfully noted that the Office Action has not made any attempt to compare claim 1615 with the cited art. Accordingly, the rejection of claim 1615 is improper and should be withdrawn.

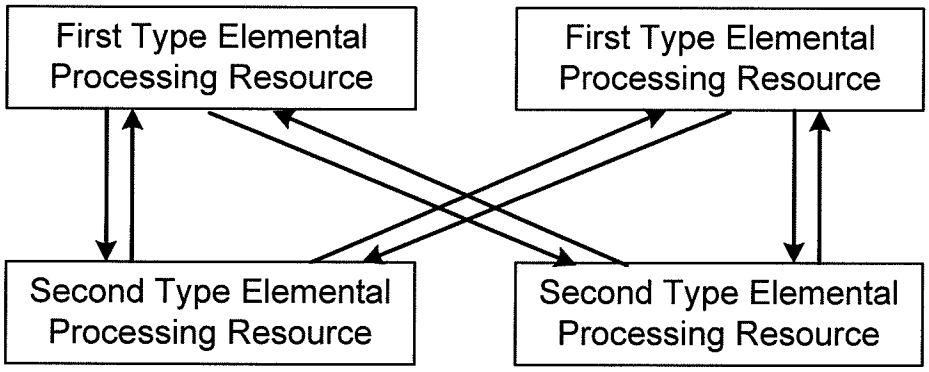
To address what also appears to be a lingering misperception about what is claimed, a general explanation about the structural configuration of the system in which the claimed methods are performed is provided. It is believed that this may will be helpful for understanding why the claims are, and have been, significantly different from the cited art.

In overview, according to the claims, the system is made up of two different types of processing resources in which there are at least two of each type. Those processing resources can each be used to execute instructions coming from a single thread or from their own threads. The instruction set that makes up all of the possible instructions that could be in a thread is such that none of the processing resources are individually capable of executing the entire instruction set (i.e. any given processing resource cannot execute at least one instruction in the set). As a result, those processing resources are referred to as "elemental" processing resources.

The first of the two different types of processing resources fetch a stream of instructions from memory and execute those instructions that they are capable of executing. In other words, the first type processing resources receive the instructions in the thread from memory. Examples of two types of processors that could individually serve as one or more of this first type elemental processing resources are the Intel 80386 or Motorola 68000.

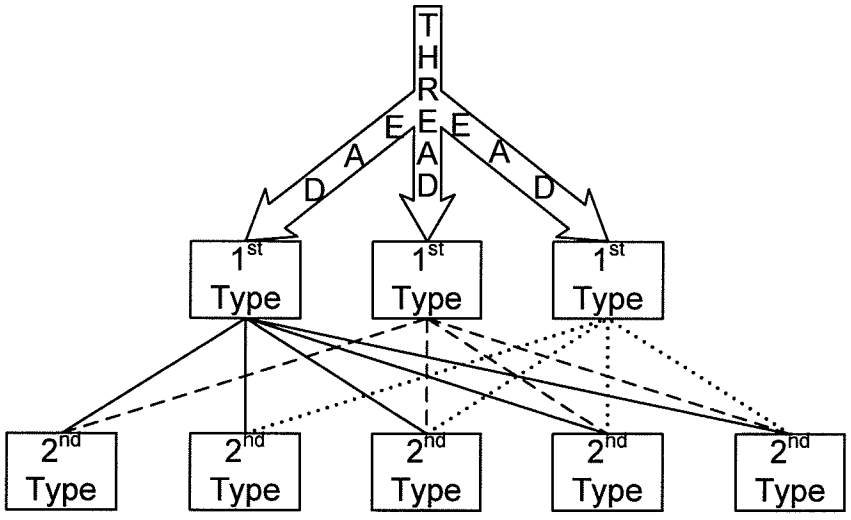
The second of the two different types of processing resources only receive instructions delegated to them by a first type processing resource and only execute those instructions delegated to them (i.e. they do not retrieve them from memory). In that regard, they are co-processors. Those second type processing resources are any of several different "flavors" of special purpose processors, for example, vector or matrix co-processors, math co-processors, encryption/decryption co-processors, delegated atomic instruction executing memory cache controllers, etc. Moreover, in any given system those second type processors can be any combination of those "flavors" (i.e. they could all be the same, all could be different, or there could be a mix and match where some are the same and others different). Examples of what could be considered floating point "flavors" of such second type elemental processing resources are the Intel 80387 or Motorola 68882 floating point co-processors.

These two different types of processing resources, as claimed, are interconnected into a unique form of configuration such that each "second type" processing resource is shared by at least two "first type" processing resources. As a result, when an instruction is encountered that one of the at least two claimed "first type" processing resources cannot execute, (as claimed) it has at least two "second type" processing resources to which it can delegate that instruction for execution and receive back a result. In the simplest case, the configurations can be conceptually represented as shown below with the lines representing the delegation and result return.



As particularly illustrated above, there are two "first type" processing resources that share the two "second type" processing resources with each other. As noted above, depending upon the particular system, each of the first type processing resources could be configured to execute instructions from individual single threads or both of the first type processing resources could be configured to execute instructions from a thread that they share in common.

In addition, as noted above, the sharing need not be balanced (i.e. there need not be any particular number of either or the same number of first type processing resources as there are second type processing resources). Rather, there could be any number of first type processing resources and second type processing resources, provided that there are at least two "first type" and at least two "second type" processing resources and at least two of the second type processing resources are shared by the first type processing resources. Thus, by way of example, a more complex configuration might conceptually look like the following.



In the example configuration above, there are three "first type" processing resources that are configured so that they can all execute instructions from a common thread. In addition, there are five "second type" processing resources, which may be of the same or different kinds. For purposes of clarity, the connections between the two types of processing resources are denoted by lines where connections from a specific "first type" processing resource are denoted by the same style line but different from the others. Moreover, for purposes of explanation, presume that the leftmost two "second type" processing resources are the same as each other and the rightmost three are each different from those two and each other. In this conceptual example then, the rightmost three "second type" processing resources are all shared by all of the "first

type" processing resources whereas the extreme leftmost "second type" processing resource is only shared by the left two "first type" processing resources and the one to its right is shared by the leftmost and extreme rightmost "first type" processing resources (i.e. the middle "first type" processing resource cannot delegate to the leftmost "second type" processing resource.

The above should provide a sufficient backdrop for understanding the claimed methods. Of course, the above conceptual configurations are only two of the infinite number of permutations and combinations that could be created to use the claimed methods.

With that background, each of the cited references can be addressed to show how each represents a significant departure from what is claimed in the instant application. Specifically, in short, all of the cited references lack a "second type" processing resource. Lacking such "second type" resources, none of them can disclose or even hint at the sharing of such resources, which is required for performing the claimed methods, whether taken alone or in combination.

Based upon the following, it will be apparent that the claimed method would not have been obvious based upon any of those references in combination with each other, absent the teachings and disclosure of the instant application.

US Patent 5,706,514 (Bonola)

Bonola describes a multiprocessor system of multiple "multimode" microprocessors. Based upon the instant application, each "multimode" microprocessor in Bonola would be best equated as an example of an elemental processing resource of the first type recited in the claims, irrespective of whether it is a master or slave. That is because, in Bonola, each microprocessor fetches and executes an instruction stream from memory and inter-processor communication among them is by software convention. Bonola does not describe anything that would be an elemental processing resource of a second type at all. As a result, Bonola does not, indeed cannot, disclose or hint at any form of method involving delegating instructions to a second type elemental processing resource, let alone a method that requires at least two first type elemental processing resources sharing at least two second type elemental processing resources.

US Patent 6,081,860 (Bridges et al.)

Bridges et al. describes a multiprocessor system consisting of multiple microprocessors that connect to multiple memory devices via a shared, arbitrated Processor Local Bus (PLB).

Each "master device" microprocessor in Bridges et al. would be best equated as an example of an elemental processing resource of a first type. Each "slave device" is a memory

controller that does not execute delegated atomic memory instructions so they are not an elemental processing resource of a second type. This is clear because each microprocessor in Bridges et al. fetches and executes an instruction stream from memory while each memory controller simply reads and writes memory, with address pipelining and arbitration. Thus, Bridges et al. does not describe anything that would be an elemental processing resource of a second type at all. As a result, Bridges et al. does not, indeed cannot, disclose or hint at any form of method involving delegating instructions to a second type elemental processing resource, let alone a method that requires at least two first type elemental processing resources sharing at least two second type elemental processing resources.

US Patent 6,907,454 (Butterworth et al.)

Butterworth et al. describes a dual processor system with a high performance master processor connected, via a bridge circuit, to a lower performance slave processor and various high and low speed memory busses.

Each processor in Butterworth et al. would be best equated as an example of an elemental processing resource of a first type, irrespective of whether it is a master or slave. This is clear because each processor in Butterworth et al. fetches and executes an instruction stream from memory and inter-processor communication is by software convention, with commands and data passed via registers and interrupt signals in bridge circuitry. Thus, Butterworth et al. does not describe anything that would be an elemental processing resource of a second type at all. As a result, Butterworth et al. does not, indeed cannot, disclose or hint at any form of method involving delegating instructions to a second type elemental processing resource, let alone a method that requires at least two first type elemental processing resources sharing at least two second type elemental processing resources.

US Patent 5,978,838 (Mohamed et al.)

Mohamed et al. describes a dual processor system on a chip where the two processors execute different instruction sets. One of the two is a general purpose processor and the other is an SIMD vector processor. Although an SIMD vector processor could be a "second type" processing resource as a general matter, that is not the case with Mohamed et al., because each has its own instruction and data caches. All they share is a main memory bus. Both the processors are full independent peer processors, executing their own software, they are not

configured as a processor and a coupled co-processor. Moreover, since Mohamed et al. is simply a dual processor system, by definition there can be no sharing.

Both processors in Mohamed et al. would be best equated as examples of an elemental processing resource of a first type, even though one is a general purpose processor and the other an SIMD vector processor. This is because each processor fetches and executes an instruction stream from memory and inter-processor communication is by software convention, with commands and data passed via registers and interrupt signals in bridge circuitry. Thus, Mohamed et al. does not describe anything that would be an elemental processing resource of a second type at all. As a result, Mohamed et al. does not, indeed cannot, disclose or hint at any form of method involving delegating instructions to a second type elemental processing resource, let alone a method that requires at least two first type elemental processing resources sharing at least two second type elemental processing resources.

Claim 144 Is Allowable

Based upon the foregoing, it should be appreciated that none of the cited references, taken alone or in any combination, disclose or suggest at any one of the following aspects of independent claim 144, let alone any combination of such aspects:

- "two or more elemental processing resources of a first type each sharing two or more elemental processing resources of a second type with the other"
- "determining whether an operation-code within the execution instruction is incapable of being executed by the one elemental processing resource of the first type and thus should be delegated to one of the shared elemental processing resource of the second type"
- "if the execution instruction should be delegated, routing the execution instruction to a shared elemental processing resource of the second type, of the at least two of that type, that is capable of executing the operation code"

Accordingly, since no combination of the cited references disclose or suggest any one or more of at least those aspects, claim 144 is allowable.

The Claims Depending From Claim 144 Are Allowable

Claims 145, 155-158, 160, 164-165, and 167-172 all depend, directly or indirectly, from claim 144, so they are all allowable for at least the same reasons.

Moreover, the dependent claims add additional aspects that are lacking from the cited references as well, taken alone or in combination. Accordingly, those dependent claims are independently allowable. For example, dependent claim 145 requires that the method be

performed "within a single processing cycle." Based upon the above, the rationale stated in paragraph 21 of the Office Action, even if true, would not address the claimed method because (ignoring the lack of a second type processor at all) a processor in Butterworth et al. does not delegate instructions to the other processors at all and performing delegation according to the claimed method in a single processing cycle is a non-trivial advance. Thus, even if there is a general desire to perform processing within the fewest clock cycles possible, that does not inexorably lead to the obviousness of doing any more than the desirability of teleportation of people from one place to another to speed up travel would render obvious an inventive process that accomplishes such teleportation based upon that well-known desire.

Dependent claims 158, 160 and 164 respectively each specify at least one specific "flavor" for one or more of the second type processing resources. Since, as demonstrated above, the cited references do not disclose or suggest any "second type" processing resources at all, they necessarily cannot disclose any such particular type.

In addition, claims 158, 160 and 164 allow for both homogeneity or heterogeneity among the second type processing resources because they require "one or more" of the at least two to be of a particular "flavor" of resource. This aspect is similarly lacking from all of the references.

Dependent claim 167 requires, while acting on instructions from an individual thread, that a first type resource to sleep while a second type resource executes an instruction from the thread delegated to it. No such operation is remotely hinted at by the cited references.

Dependent claim 168 specifically require the resources "dynamically to turn on and off to maintain a desired level of power draw while maximizing processing throughput." The Office Action-cited part of Bonola does not begin to disclose or suggest this aspect. All Bonola discloses is that a processor can "sleep," a far cry from regulating sleep in the manner claimed.

Similarly, Bonola's simple disclosure of a processor that can sleep does not begin to disclose or suggest the subject matter of claims 169 and 170 which respectively require "generating an execution-instruction signal from at least one of the elemental processing resources, wherein the execution-instruction signal from the elemental processing resources themselves shuts off processing resources while idling" and "generating an execution-instruction signal from at least one of the elemental processing resources, wherein an execution-instruction signal from processing resources themselves turn on processing resources when execution-instruction signal processing is required."

Accordingly, the above-referenced dependent claims are all allowable for those independent reasons.

Claim 1615 Is Allowable

First, as noted above, the Office Action does not even attempt to compare any of the references to claim 1615. As such, the rejection of that claim should be withdrawn.

Nevertheless, none of the cited art, alone or in combination, disclose claim 1615 because they neither disclose nor suggest at any one of the following aspects of independent claim 144, let alone any combination of such aspects:

- "at least two IPU type processing resources"
- "two or more elemental processing resources of an other type comprising, in any combination, one or more of either an MPU type, an instruction delegating cache type or a vector type"
- " the elemental processing resources being configured such that a first and a second of theIPU type processing resources each share at least two of the other type elemental processing resources"
- "delegating the other execution-instruction from the thread to one of the two or more elemental processing resources of the other type that is shared between the first IPU and a second IPU"
- "maintaining the thread in the first IPU in a sleep state until an indicator that the processing of the other execution-instruction from the thread by the other type processing resource is returned."

Accordingly, claim 1615 would not have been obvious in view of any of the cited references.

CONCLUSION

It is respectfully submitted that the above demonstrates that all of the pending claims would not have been obvious in view of the references cited by the Office Action, whether taken alone or in combination. Accordingly, it is respectfully submitted that all of the pending claims are allowable and early, favorable action in that regard is respectfully requested.

In the event that any claims are still not allowable after considering this paper on the merits, particularly if based upon the above cited references, applicant respectfully requests an interview with the Examiner to discuss any lingering Examiner concerns or issues.

AUTHORIZATION

The Commissioner is hereby authorized to charge any fees which may be required for consideration of this Amendment to Deposit Account No. **504827**, Order No. **1004437-006US**.

Although no extension is believed necessary, in the event that an additional extension of time is required, the Commissioner is requested to grant a petition for that extension of time which is required to make this response timely and is hereby authorized to charge any fee for such an extension of time or credit any overpayment for an extension of time to Deposit Account No. **504827**, Order No. **1004437-006US**.

Respectfully submitted,
Locke Lord Bissell & Liddell, L.L.P.

Dated: November 3, 2009

By: 

Richard Straussman
Registration No. 39,847

Correspondence Address:
Associated with Customer No. **85775**